HAL
archives-ouvertes.fr

A new distance measure based on the exchange operator for the HFF-AVRP

Marie-Eleonore Marmion, Laetitia Jourdan, Clarisse Dhaenens

▶ **To cite this version:**

Marie-Eleonore Marmion, Laetitia Jourdan, Clarisse Dhaenens. A new distance measure based on the exchange operator for the HFF-AVRP. [Intern report] RR-7263, INRIA. 2010. <inria-00475710>

HAL Id: inria-00475710
https://hal.inria.fr/inria-00475710

Submitted on 22 Apr 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A new distance measure based on the exchange operator for the HFF-AVRP

Marie-Éléonore Marmion — Laetitia Jourdan — Clarisse Dhaenens

N° 7263

Avril 2010

Optimization, Learning and Statistical Methods

ISSN 0249-6399 ISRN INRIA/RR--7263--FR+ENG

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE
INRIA
centre de recherche LILLE - NORD EUROPE

A new distance measure based on the exchange operator for the HFF-AVRP

Marie-Éléonore Marmion * , Laetitia Jourdan * , Clarisse Dhaenens *

Theme : Optimization, Learning and Statistical Methods
Équipe-Projet DOLPHIN

Rapport de recherche n° 7263 — Avril 2010 — 14 pages

Abstract: The Heterogeneous Fixed Fleet Asymmetric Vehicle Routing Problem (HFF-AVRP) is a $\mathcal{NP}$-hard optimization problem. Instances analysis and in particular, fitness landscape analysis, may help problem solving. Such analysis require the definition of a distance between feasible solutions. Such a distance does not exist for the HFF-AVRP and this report aims at proposing a new distance measure defined from the exchange operator. In order to compute the exchange-distance between two solutions, four algorithms are suggested and then experimented. One of them is proved to be robust and to give the exact distance whereas others only compute an upper bound.

Key-words: distance, HFF-AVRP

* LIFL, CNRS, Université Lille 1, INRIA

Centre de recherche INRIA Lille – Nord Europe
Parc Scientifique de la Haute Borne
40, avenue Halley, 59650 Villeneuve d'Ascq
Téléphone : +33 3 59 57 78 00 — Télécopie : +33 3 59 57 78 50

Une nouvelle mesure de distance basée sur l'opérateur d'échange pour le HFF-AVRP

Résumé : Le problème asymétrique de tournée de véhicules avec une flotte fixe hétérogène (HFF-AVRP) est un problème d'optimisation $\mathcal{NP}$-difficile. L'analyse des instances et en particulier, l'analyse des paysage de fitness peut aider à résoudre le problème. Pour réaliser une telle analyse, il est nécessaire d'avoir une distance exacte entre les solutions. Une telle distance n'existe pas pour le HFF-AVRP, dans ce rapport, nous proposons une nouvelle mesure de distance définie à partir de l'opérateur d'échange. On propose et teste quatre algorithmes pour calculer cette distance d'échange entre deux solutions. On monte que seulement l'un d'entre eux est robuste et donne la distance exacte alors que les autres nous donnent uniquement une borne maximale.

Mots-clés : distance, HFF-AVRP

1 Introduction

The Capacited Vehicle Routing Problem (CVRP) defines a category of problems often met by transportation companies. It consists in satisfying a set of customer demands thanks to a fleet of vehicles subject to limited capacity, trying to minimize costs due to road travelled. The Heterogenous Fixed Fleet Asymmetric Vehicle Routing Problem (HFF-AVRP) has been proposed [1] as a realistic extension of the basic Vehicle Routing Problem where a fleet with a fixed number of vehicles of various types with different capacities and variable unit running costs is considered.

The HFF-AVRP may be defined as follows: let $G = (V, A)$ be a complete directed graph, where $V = (v_0, v_1, v_2, \ldots, v_{nc})$ is the node set and $A = \{(v_i, v_j) : v_i, v_j \in V\}$ is the arc set. Vertex v_0 corresponds to the depot and vertices v_1 to v_{nc} correspond to the nc customers. A fleet of heterogeneous (different types) vehicles (total of nv vehicles) uses the depot as a starting base. Let us denote by c_t the variable cost per distance unit of a vehicle of type t and by Q_t its capacity. The number of vehicles of each type is limited (fixed). Each vehicle can be allocated at most to one route. A non-negative demand q_i is associated with vertex v_i ($i = 1..nc$) corresponding to the customers. A non-negative distance d_{ij} ($d_{ii} = 0$ for all $i = 1..nc$) is associated with each arc $(v_i, v_j) \in A$, $v_i \neq v_j$, of the graph G and there exists at least two different i, j in $1..nc$ such that $d_{ij} \neq d_{ji}$ (the distance matrix is asymmetric). This problem aims at determining a set of vehicle routes, each starting and ending at the depot, such that a single vehicle supplies each customers demand, the total demand of customers assigned to each route does not exceed the capacity of the vehicle assigned to it, and the total cost is minimized.

HFF-AVRP is a $\mathcal{NP}$-hard combinatorial optimization problem where it is very difficult for any resolution method to find a global optimum. With each HFF-AVRP instance, a structure, named landscape [2], can be associated. It corresponds to a triplet (Ω, $\mathcal{N}$, f) where Ω is a search space, $\mathcal{N}$ is a neighborhood definition and f a fitness evaluation. Landscape analysis helps to find particularities of instances which may have influences on the resolution and so give appropriate information to the decision maker.

Some landscape analysis studies [3, 4, 5] have been made for different optimization problems, and they use several indicators to find interesting properties. A lot of indicators require to define a distance between feasible solutions. This distance should evaluate the number of elementary moves required to move from one solution to another. Hence, the distance may be different regarding the operator used. So, in order to make a landscape analysis of HFF-AVRP instances, a distance between solutions is required. As far we know, up to now, no exact distances have been proposed for such problems. Our study aims at filling this gap giving a reliable distance between HFF-AVRP solutions as well as an algorithm to compute it efficiently. Section 2 proposes a representation and a neighborhood operator to HFF-AVRP solutions. Section 3 presents the distance and several algorithms to compute it. In section 4, experimentations of the different proposed algorithms are lead. Finally, section 5 draws a conclusion and exposes perspectives for this work.

2 An HFF-AVRP structure

The HFF-AVRP is less studied than the symmetric CVRP with a homogeneous fleet. The dedicated literature (for example [6, 7]) does not deal with neither solution representation nor neighborhood definition. As studies on CVRP are numerous, they give a reliable basis for an HFF-AVRP study. Figure 1 presents an HFF-AVRP solution with 7 customers and 4 vehicles. This solution contains four routes: the first one, allocated to vehicle 1, visits customer 1, then customer 3 and finishes with customer 2. The second route, allocated to vehicle 2, visits customer 4, then visits customer 7 to finish with customer 5. The thirs one, allocated to vehicle 3, only visits customer 6. The last one is empty. This figure illustrates the key concepts of the current section.

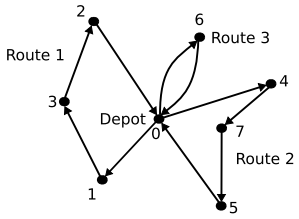

Figure 1: Example of an HFF-AVRP solution with 7 customers and 3 vehicles.

2.1 State of the art on solution representation

Solving a combinatorial optimization problem highly depends on how it has been modelled. For CVRP, different representations may be found in the literature. In 1983, Beasley [8] introduces the *big tour* representation. Customers are listed in the order of their visit, and then a split procedure forms the routes : it is *route first / cluster second* principle. Thus, a solution corresponds to the route list in which customers are organised by vehicle visit order. For example, the HFF-AVRP solution of Figure 1 could be first represented as 1 3 2 4 7 5 6 and then the split procedure would give the representation 1 3 2, 4 7 5, 6, $\emptyset$.

Kubiak [9] prefers to consider a set of routes where each route is itself an ordered set of customers. This representation is the most used because it allows to easily access any route and then any customer. For example, the HFF-AVRP solution of Figure 1 would be represented as [[1 3 2], [4 7 5], [6], []].

Another representation [10], named adjacency representation, put the customers in a vector whose size corresponds to the number of customers. Each index (numbered from 1 to *nc*) represents a customer. To each index corresponds its successor. When a customer finishes a route, the vector is filled with a zero at the correponding index. For example, the HFF-AVRP solution of Figure 1 is represented with the vector $[\ \overset{1}{3}\ \overset{2}{0}\ \overset{3}{2}\ \overset{4}{7}\ \overset{5}{0}\ \overset{6}{0}\ \overset{7}{5}\ ]$ (indexes are added to help the understanding).

2.2 Distance and neighborhood operator

In the literature, three different ways to compute the distance between CVRP solutions exist (these distances are detailed in part 3.1.):

1. distance with structural comparisons [9],
2. approximative distances such as Hamming distance [11] and pseudo-distance [12],
3. distance defined by an operator [13].

In a context of landscape analysis, the third type of distance is particularly interesting as the concepts of neighborhood and distance have to be linked.

The exchange operator (also called interchange operator) is a classical operator for permutation problems (see Schiavinotto and Stützle [4]). It consists in switching two different clients. For a permutation of size N, this operator (i, j) exchanges element i with element j. In the traveling salesman problem (TSP), this operator is often used. Since a VRP can be viewed as a multiple TSP, we choose to build the neighborhood structure using the exchange operator.

2.3 A solution representation adapted to a neighborhood operator

Even if representations of the literature are widely used for different approaches of CVRP resolution, they are maladaptive to work with the HFF-AVRP and the exchange operator. Therefore, the chosen representation for this study consists in a list of customers (numbered between 1 and nc, the number of customers) ranked following their visit order and separated by zeros meaning the end of a route and the start of another. For readability, a solution starts and finishes with a zero. In the following, indexation is from 0 to $nc + nv - 1$. For example, the HFF-AVRP solution of Figure 1 could be expressed as:

$$(0\ 1\ 3\ 2\ 0\ 4\ 7\ 5\ 0\ 6\ 0\ 0)$$

With this representation, it is easy to know in which route a customer is and its visiting order.

This representation suits to the exchange operator which consists in exchanging either two customers or one customer and a zero (exchanging two zeros is forbidden otherwise the solution does not change). So, with the previous example, the exchange operator $(1, 7)$ acts on customers located at indexes 1 and 7 and gives the solution:

$$(0\ 5\ 3\ 2\ 0\ 4\ 7\ 1\ 0\ 6\ 0\ 0)$$

Notice that this solution can become unfeasible after an exchange if the capacity constraints of vehicles are not respected. For this representation and its associated operator, we aim to define an associated distance.

3 An adaptated distance

3.1 State of the art

Only few studies exist on HFF-AVRP, so we have studied distances proposed for the CVRP. These distances can be divided into three groups:

Comparative distance Kubiak, in a *fitness-distance* analysis [9] uses three distances that can be described as comparative. Indeed, he measures the similarities between two solutions by examining either if customers have the same predecessor and successor in the two solutions, or if each route of the two solutions visits the same customers. Even if these distances are not based on a specific operator, they can help to know how dissimilar or close two CVRP solutions are.

Approximate distance Hamming distance is a well-known distance in combinatorial optimization. For a path-relinking study, Ho and Gendreau [11] adapt this distance to CVRP solutions by numbering all edges (an edge is the road between customer/vehicle to its successor) and creating a bit-vector. Hamming distance corresponds to the number of different bits between two solutions. Another approximate distance is proposed by Prins and al. [12] in another path-relinking study. This distance consists in giving a weight to each edge of the first solution according to the similarity or the difference in the second solution.

Operator related distance In a study on path relinking for the CVRP, Sörensen [13] presents the *edit-distance* based on three elementary operators: add, remove and substitute. The distance between two feasible solutions is the minimal number of operator applications required to move from the first solution to the second one. Then, he tranforms this distance keeping only add and remove operators. Since solutions are represented as a *big-tour*, this distance corresponds to the insert distance for permutation, defined in [4]. This distance is an interesting proposition of a distance between CVRP solutions defined from an operator.

Regarding all these distances, we see that comparative and approximate distances give information about closeness or dissimilarities between solutions, their edges and routes, whereas distances based on operators give a measure that can be considered through the search space with a neighborhood structure. As explained before, to make a landscape analysis, an operator based distance is needed.

3.2 The proposed exchange-distance

The interest of having a distance associated to an operator is that the number of modifications required to move from a solution to another can be measured and evaluated. In this study, we aim at providing a distance associated to the exchange operator. Our proposed distance, called exchange-distance d_e, is defined by the minimal numbers of exchange operators to be applied to move from an HFF-AVRP solution to another. This definition verifies the mathematical conditions:

- positive definiteness is clear,

$$d_e \geq 0$$

$$\forall (s_1, s_2), \quad d_e(s_1, s_2) = 0 \Leftrightarrow s_1 = s_2$$

- the symmetry property comes from the symmetry property of the exchange operator
$$\forall(s_1, s_2), \quad d_e(s_1, s_2) = d_e(s_2, s_1)$$
- and the triangle inequality is given by the minimal caracteristic of the definition
$$\forall(s_1, s_2, s_3),$$
$$d_e(s_1, s_2) \leq d_e(s_1, s_3) + d_e(s_3, s_2)$$

Indeed, if p exchange operator applications are needed to move from s_1 to s_3 and q exchange operator applications are needed to move from s_3 to s_2, then it is possible to move from s_1 to s_2 by at most $p+q$ exchange operator application.

With this distance, the neighborhood of an HFF-AVRP solution defined with the exchange operator can be described as the set of all the HFF-AVRP solutions which are at a distance equal to one. In the same way, the n-neighborhood of an HFF-AVRP solution is the set of HFF-AVRP solutions which are at a distance equal to n from the first one. In the case of an HFF-AVRP instance with nc customers and nv vehicles, we can proove that the maximal neighborhood is nc-order (it does not depend on the number of vehicles).

3.3 Algorithms to compute the exchange-distance

Defining and computing a distance is the aim of this study. So, an algorithm which computes the exchange-distance between HFF-AVRP solutions s_1 and s_2 is required. Obviously, this algorithm depends on the HFF-AVRP representation. In the following, we present four algorithms of different complexity and accuracy.

***Permutation-adapted* algorithm** As the representation looks like a permutation, we first try to use an existing algorithm which calculates the exchange-distance in the case of permutations. Bachelet [3] proposed an $\mathcal{O}(N)$ algorithm for such purpose (N is the permutation size). This algorithm, based on the inverse operator of the exchange operator for a permutation, has to be adapted to our representation. Since zeros appear several times in our representation, we decide to number vehicles from $nc+1$ to $nc+nv$ in order to be able to use Bachelet algorithm. Thus, we obtain a real permutation. This adapted algorithm, called *permutation-adapted*, is fast. Unfortunately having differenciated vehicles leads to wrong computed distance. Even if it gives an upper bound of the real value of the distance, a robust calculation is needed. We target to find another one. Three algorithms emerge from this work.

***Simple greedy* algorithm** This second strategy is greedy *i.e.* solution s_1 is scanned from left to right in order to transform it into s_2 with the minimal number of applications of the exchange operator. At the end of the p-step, the p first elements (customers and vehicles) in the two considered solutions are the same. So, at each step, if the values in s_1 and s_2 are different, the value of s_2 is sought through s_1 from index $p+1$. And then, the exchange operator is called. This algorithm computes an upper bound as a feasible sequence of exchange operators is produced.

***With-choice greedy* algorithm** The *simple greedy* algorithm has a major drawback: it does not optimize the replacement of zeros. A better strategy needs to be defined. If a zero has to be replaced, instead of choosing the first one in s_1, the idea is to try to find a zero which puts the element p at its position . This new algorithm is more efficient than the previous one when the exchange operator puts both vehicle and customer in their right place. The calculation of the distance becomes more precise *i.e.* the value of the upper bound is ameliorated.

***On-hold greedy* algorithm** Ordering solution at each step introduces errors which increase the distance. We decided to break this approach by ordering all the customers first. We realize all the zeros are then well placed back. In order to have an optimized algorithm, a memory with a list of indexes is created. When a zero has to be replaced and when any exchange allows the current customer to be replaced at his right position, the index of the step is put in the memory. Hence, the memory contains all the indexes ever scanned where s_1 values are not equal to s_2 ones. This list has to be scanned at each step because a customer refered by it can be found. Using a classical exchange procedure, the *on-hold greedy* algorithm is detailed in Algorithm 1.

The following example illustrates the algorithm. To move from s_1 to s_2, four exchange operations are needed. Let us see the detailed execution of the algorithm:

$$s_1 : \ (0\ 1\ 3\ 2\ 7\ 0\ 6\ 0\ 4\ 5\ 0)$$

$$s_2 : \ (0\ 1\ 3\ 2\ 0\ 4\ 7\ 5\ 0\ 6\ 0)$$

- $i = 1$ to 3
  $s_{1i} = s_{2i}$: the values are equal in both solutions.
  $distance = 0$, $list = ()$.

- $i = 4$
  $s_{1i} \neq s_{2i}$: 0 is needed. $list$ is empty. $s_{2i} = 0$: no 0 at the right position of 7.
  Index 4 is put in the memory $list$.
  $distance = 0$, $list = (4)$.

- $i = 5$
  $s_{1i} \neq s_{2i}$: 4 is needed. It is not refered by the $list$.
  4 is found at index 8.
  Exchange $(5, 8)$ gives (0 1 3 2 7 4 6 0 0 5 0).
  $distance = 1$, $list = (4)$.

- $i = 6$
  $s_{1i} \neq s_{2i}$: 7 is needed. It is refered by the $list$.
  $s_{1i} \neq 0$: the $list$ remain unexchange. Exchange $(4, 6)$ gives (0 1 3 2 6 4 7 0 0 5 0).
  $distance = 2$, $list = (4)$.

- $i = 7$
  $s_{1i} \neq s_{2i}$: 5 is needed. It is not refered by the $list$.
  5 is found at index 9.
  Exchange $(7, 9)$ gives (0 1 3 2 6 4 7 5 0 0 0).
  $distance = 3$, $list = (4)$.

- $i = 8$
  $s_{1i} = s_{2i}$: the values are equal in both solutions.
  $distance = 3$, $list = (4)$.

- $i = 9$
  $s_{1i} \neq s_{2i}$: 6 is needed. It is refered by the $list$.
  $s_{1i} = 0$: the index which refers 6 is removed from the $list$. Exchange $(4, 9)$ gives (0 1 3 2 0 4 7 5 0 6 0).
  $distance = 4$, $list = ()$.

This algorithm needs more conditions than the other two greedy algorithms. Experiments reported later show it always finds the exact distance.

Algorithm 1 On-hold greedy algorithm

Input: s_1, s_2, N
Output: ExchangeDist(s_1,s_2)

$distance \leftarrow 0$
$list \leftarrow ()$
temp {variable to keep the index if exchange is possible}
for $i = 1$ to N **do** {where N is the length of the solutions s_1 and s_2}
  if $s_{1i} \neq s_{2i}$ **then**
    if $list$ is not empty and it exists j such that $s_{1list_j} = s_{2i}$ **then** {s_{2i} is refered in $list$}
      temp $\leftarrow list_j$
      if $s_{1i} = 0$ **then** {0 finds its positions, index has to be removed from the $list$}
        $list \leftarrow list \setminus list_j$
      end if
    else
      if $s_{2i} = 0$ **then**
        if It exists j such that $s_{1j} = 0$ and $s_{1i} = s_{2j}$ **then** {s_{1i} can be directly well repositioned}
          temp $\leftarrow j$
        else
          $list \leftarrow list :: i$
        end if
      else {It exists j such that $s_{1j} = s_{2i}$ *i.e.* $s_{2i} \neq 0$ and $j \notin list$}
        temp $\leftarrow j$
      end if
    end if
    if temp exists **then**
      Exchange(s_1,i,temp)
      $distance \leftarrow distance + 1$
    end if
  end if
end for

On the algorithmic point of view, the three proposed *greedy* algorithms are more and more difficult to implement. Indeed, more and more conditions are

defined in order to be closer to the exact distance. On the complexity point of view, they are equivalent. Indeed, they have on average an $\mathcal{O}(N^2)$ complexity (N is the length of the solution). Therefore, the *on-hold greedy* algorithm has to be considered to compute the distance between two HFF-AVRP solutions, because it gives the exact distance and it is not more time consuming than the other two.

4 Experimentations

The *permutation-adapted* algorithm and the three *greedy* algorithms are tested on the proposed HFF-AVRP representation in order to find cases where the calculation of the distance between two solutions is exact. Here, a solution is represented by a vector of N elements with nc customers and $nv + 1$ zeros (as previously said, nv is the number of vehicles). Note, the number of vehicles taken into account is smaller than half the number of customers, to avoid as possible, routes with only one customer to visit.

4.1 Implementation

Tests consist in randomly creating solutions at a d chosen distance from a random initial solution. Then each algorithm computes the distance between the initial and the final solutions to compare it with the exact value (d).

Initialization To generate HFF-AVRP solutions, every customers are ranked in the numerical order (from 1 to nc) in a vector. Then, two zeros are added at the beginning and at the end of the vector. They cannot be exchanged because the representation has to start and to finish by zero. To have nv routes, $nv - 1$ zeros are randomly inserted in the vector in such a way that no route is empty (zero can not be closed).

Successive neighborhood From this initial generated solution, the exchange operator is applied d times to get a solution which belongs to its d-neighborhood. Remember the maximal neighborhood of a solution is the number of customers (nc). Two cases can be distinguished to create a d-neighborhood. Indeed, if d is equal to nc, as in permutation problems, a cycle is sufficient to have a solution in the d-neighborhood (order d). Since no route is empty, it is impossible to have a zero which replaces another. This is why we are sure of the existence of a nc-neighbor. For all d-neighborhood with d strictly lower than nc, a list is created with all the indexes between 2 and $(N-1)$ corresponding to the elements which could be exchanged. Two indexes are chosen in this list (corresponding to at most one zero). If the indexes correspond to two customers then the first index is removed from the list. Thus, during next steps, the first element will not be put in its first position because it does not belong to the list anymore and so for the second element, because it cannot move anymore to its first position. If the two chosen indexes corresponds to a vehicle and a customer, then the two indexes are removed from the list: for sure any zero will be able to substitute another. At the end of step $d = nc - 1$, the list is empty.

Experiments The four algorithms are tested using several parameters:

- number of customers,
- number of vehicles,
- distance of the neighborhood.

Indeed, the larger is the neighborhood and the more the number of customers and/or vehicles are/is huge, the bigger is the probability to have a wrong computed distance. Algorithms are tested with 10, 100 and 1000 customers. For each case the number of vehicles equals: 1 vehicle (case of the traveling salesman problem), then 5%, 20% and 50% of the number of customers. The $nc/8$, $nb/4$, $nc/2$, $nc-1$ and nc -neighborhood are computed for each case. Table 1 reports the chosen values for the tests.

Number of Customers	Number of Vehicles	Neighborhood order
10	1, 2, 5	1, 2, 5, 9, 10
100	1, 5, 20, 50	12, 25, 50, 99, 100
1000	1, 50, 200, 500	125, 250, 500, 999, 1000

Table 1: Tested values.

Each possibility has been tested 1000 times.

4.2 Analysis

To realize the analysis over the thousand computed values, two indicators are used. The first one is the error percent (1) which gives information about the difference between a theoretical value and its computed one. The error percent for each case could be computed as:

$$Err\% = \frac{\overline{d-D}}{D} \tag{1}$$

where d is the computed distance and D is the theoretical distance. As the four algorithms compute an upper bound, d is positive and so $\overline{d-D}$, the average of the difference, is meaningful.
The second indicator is the coefficient of variation of a data series (2) which is dimensionless and gives information about dispersion of a distribution in comparison to its mean. The coefficient of variation is computed as:

$$\Delta = \frac{\sigma}{\bar{d}} \tag{2}$$

where σ is the standard deviation.

Table 2 presents the error percents for each algorithm and the computed distances. The values are given in three different vertical groups to show the efficiency of each algorithm depending on the number of customers. Moreover, the values are split into three horizontal groups in order to show the influence of the number of vehicles.

	10 customers				100 customers				1000 customers			
Algorithms	1	2	3	4	1	2	3	4	1	2	3	4
Neighborhood 0rder	1 vehicle				5 vehicles				50 vehicles			
nc/8	0	0	0	0	6.19	3.68	0.36	0	28.76	22.31	1.88	0
	(0)	(0)	(0)	(0)	(7.42)	(6.53)	(2.18)	(0)	(3.74)	(6.73)	(3.9)	(0)
nc/4	0	0	0	0	4.79	3	0.43	0	15.95	13.66	2.64	0
	(0)	(0)	(0)	(0)	(3.92)	(3.73)	(1.57)	(0)	(1.32)	(2.57)	(2.43)	(0)
nc/2	0	0	0	0	3.34	2.46	0.91	0	8.58	8.01	3.13	0
	(0)	(0)	(0)	(0)	(1.75)	(1.87)	(1.49)	(0)	(0.43)	(0.74)	(1.23)	(0)
nc-1	0	0	0	0	1.94	1.77	1.14	0	4.45	4.39	3.12	0
	(0)	(0)	(0)	(0)	(0.79)	(0.85)	(0.93)	(0)	(0.17)	(0.2)	(0.42)	(0)
nc	0	0	0	0	3	0	0	0	4.8	0	0	0
	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)
	2 vehicles				20 vehicles				200 vehicles			
nc/8	0	0	0	0	80.57	54.81	2.94	0	136.24	120.23	9.43	0
	(0)	(0)	(0)	(0)	(19.35)	(26.27)	(11.51)	(0)	(4.55)	(8.79)	(10.95)	(0)
nc/4	0	0	0	0	49.44	36.89	4.84	0	72	66.2	11.38	0
	(0)	(0)	(0)	(0)	(7.46)	(11.77)	(8.51)	(0)	(1.91)	(4.03)	(5.97)	(0)
nc/2	0	0	0	0	28.26	24.45	7.6	0	37.54	36	12.41	0
	(0)	(0)	(0)	(0)	(3.07)	(4.63)	(5.53)	(0)	(0.61)	(1.25)	(2.78)	(0)
nc-1	0	0	0	0	15.56	15.12	10.52	0	19.32	19.21	13.82	0
	(0)	(0)	(0)	(0)	(1.24)	(1.43)	(2.3)	(0)	(0.17)	(0.22)	(0.71)	(0)
nc	0	0	0	0	18	0	0	0	19.8	0	0	0
	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)
	5 vehicles				50 vehicles				500 vehicles			
nc/8	51.5	27.4	0	0	263.83	198.59	8.47	0	357	323.17	20.32	0
	(58.96)	(55)	(0)	(0)	(17.98)	(31.32)	(26.36)	(0)	(4.54)	(9.44)	(20.13)	(0)
nc/4	44.65	23.5	0.8	0	147.32	119.99	11.3	0	185.5	174.53	21.79	0
	(34.52)	(34.14)	(7.96)	(0)	(8.58)	(16.2)	(17.2)	(0)	(2.22)	(4.7)	(9.87)	(0)
nc/2	29.66	22.08	3.7	0	80.75	72.71	16.33	0	95.46	92.42	23.45	0
	(14.71)	(16.44)	(10.15)	(0)	(3.95)	(6.76)	(9.99)	(0)	(0.93)	(1.9)	(4.51)	(0)
nc-1	19.93	16.91	7.52	0	44.01	42.65	22.28	0	48.91	48.44	26.01	0
	(7.88)	(8.43)	(8.5)	(0)	(1.36)	(1.9)	(4.14)	(0)	(0.24)	(0.43)	(1.41)	(0)
nc	30	0	0	0	48	0	0	0	49.8	0	0	0
	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

Table 2: Error percents of the distance and coefficients of variation for each measure of the experimentation in brackets. Algorithm 1 stands for *permutation-adapted* algorithm, 2 for *simple greedy* algorithm, 3 for *with-choice greedy* algorithm and 4 for *on-hold greedy* algorithm.

Undoubtedly, the *on-hold greedy* algorithm can be considered as robust. Indeed, the error between the exact and the computed values is always null. Besides, Table 2 presents in brackets the coefficients of variation which give information about the behaviour of the other three algorithms. Indeed, regarding only error percents, the *on-hold greedy* algorithm is the best and the *with-choice greedy* algorithm is better than the *simple greedy* algorithm which is better than the *permutation-adapted* algorithm.

For these three last algorithms, the larger the number of customers and/or the number of vehicles are, the higher the error percent is. The *permutation-adapted* algorithm and the *simple greedy* algorithm tend to be confused as soon as these numbers increase whereas the *with-choice greedy* algorithm seems not to be perturbed too much when the number of customers increases because values remain low for 100 or 1000 customers. However, the coefficient of variation gives an interesting information about how different or not the thousand values for each case of the experimentation are. Even if the results of the *permutation-adapted* algorithm and the *simple greedy* algorithm are closed and often bad, the coefficient of variation is always greater for the last one. The chance to give a better value is huge. Also, we can see that the *with-choice greedy* algorithm has got, in the case of 100 and 1000 customers, a coefficient of variation greater than the *permutation-adapted* algorithm and the *simple greedy* algorithm ones; it shows that this algorithm, even if is better, the chance to get a wrong value is sizeable.

Let us remark, for 1-vehicle (*i.e.* travelling salesman problem), all algorithms are robust, and so, *permutation-adapted* algorithm is the fastest one.

5 Conclusion and perspective

This study aims to fill a gap by defining a distance between HFF-AVRP solutions and an algorithm to compute it. Therefore, a solution representation and a neighborhood operator are needed. A representation close to the permutation representation and the exchange operator were chosen. Then a distance, called the exchange-distance, was defined by the exchange operator as the minimal number of exchanges necessary to move from an HFF-AVRP solution to another. With this theoretical definition of the exchange-distance, a robust algorithm was searched to compute it. Four algorithms were suggested and tested. Experimentations showed that the *on-hold greedy* algorithm computes the exact distance, and so it could be used in order to work on distance between the HFF-AVRP solutions. Since this study gives a reliable distance based on the exchange operator between feasible HFF-AVRP solutions, the next step is to realize a landscape analysis using it.

References

[1] E. Taillard, "A heuristic column generation method for the heterogeneous vrp," *RAIRO, Operations Research*, vol. 33, no. 1, pp. 1–14, 1999.

[2] S. Wright, The roles of mutation, inbreeding, crossbreeding and selection in evolution, in: D. Jones (Ed.), Proceedings of the Sixth International Congress on Genetics, Vol. 1, 1932.

[3] V. Bachelet, Métaheuristiques parallèles hybrides : application au problème d'affectation quadratique, Ph.D. thesis, Université des Sciences et Technologies de Lille (1999).

[4] T. Schiavinotto, T. Stützle, A review of metrics on permutations for search landscape analysis, Computers and Operations Research 34 (2007) 3143–3153.

[5] S. Verel, G. Ochoa, M. Tomassini, The connectivity of NK landscapes' basins: a network analysis, in: Proceedings of Artificial Life Conference Alife XI, 2008, pp. 648–655.

[6] G. Laporte, H. Mercure, Y. Nobert, An exact algorithm for the asymmetrical capacitated vehicle routing problem, Networks 16 (1986) 33–46.

[7] P. Toth, D. Vigo, An overview of vehicle routing problems, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001, pp. 1–26.

[8] J. Beasley, Route-first cluster-second methods for vehicle routing, Omega 11 (1983) 403–408.

[9] M. Kubiak, Distance measures and fitness-distance analysis for the capacitated vehicle routing problem, Vol. 39 of Operations Research/Computer Science Interfaces Series, Springer US, 2007, Ch. 18, pp. 345–364.

[10] Z. Michalewicz, D. B. Fogel, How to Solve It: Modern Heuristics, Springer-Verlag, 2000.

[11] S. Ho, M. Gendreau, Path relinking for the vehicle routing problem, Journal of Heuristics 12 (2005) 55–72.

[12] C. Prins, C. Prodhon, R. Wolfer Calvo, Solving the capacitated location-routing problem by a GRASP complemented by a learning process and a path relinking, A Quarterly Journal of Operations Research 4 (2006) 221–238.

[13] K. Sörensen, Path relinking for the vehicle routing problem using the edit distance, in: The 6th metaheuristics international conference, MIC2005, Vienna, Austria, 2005, pp. 839–846.

Centre de recherche INRIA Lille – Nord Europe
Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399